

AD-A181 226

INTEGRATED INFORMATION SUPPORT SYSTEM (IISS) VOLUME 3
IISS CONFIGURATION (U) GENERAL ELECTRIC CO SCHENECTADY
NY PRODUCTION RESOURCES CONSU N MARVIN 01 NOV 85

1/1

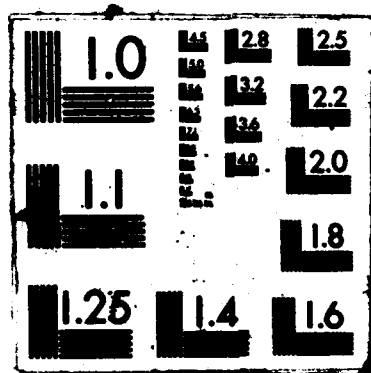
UNCLASSIFIED

CMA620124000 AFMIL-TR-86-4006-VOL-3-PT-7

F/G 12/5

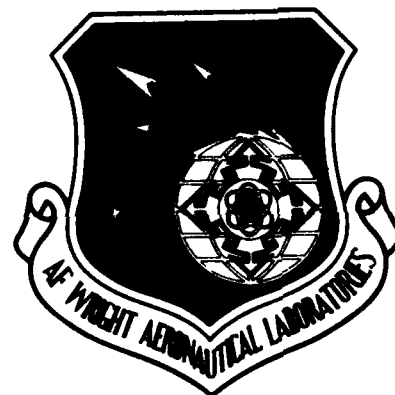
NL





**AFVAL-TR-86-4006
Volume III
Part 7**

AD-A181 226



**INTEGRATED INFORMATION
SUPPORT SYSTEM (IISS)
Volume III - IISS Configuration Management
Part 7 - SCM Administrator's Manual**

**General Electric Company
Production Resources Consulting
One River Road
Schenectady, New York 12345**

**Final Report for Period 22 September 1980 - 31 July 1985
November 1985**

Approved for public release; distribution is unlimited.

PREPARED FOR:

**MATERIALS LABORATORY
AIR FORCE WRIGHT AERONAUTICAL LABORATORIES
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AFB, OH 45433-6535**

**DTIC
ELECTE
JUN 16 1987
S D
E**

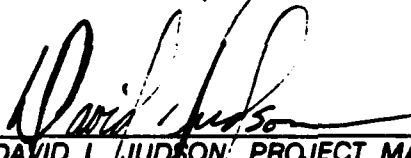
87 6 12 124

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

This report has been reviewed by the Office of Public Affairs (ASD/PA) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.




DAVID L. JUDSON, PROJECT MANAGER
AFWAL/MLTC
WRIGHT PATTERSON AFB OH 45433

5 Aug 1986

DATE

FOR THE COMMANDER:



GERALD C. SHUMAKER, BRANCH CHIEF
AFWAL/MLTC
WRIGHT PATTERSON AFB OH 45433

7 Aug 86

DATE

"If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify AFWAL/MLTC, W-PAFB, OH 45433 to help us maintain a current mailing list."

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

1 November 1985

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified		1b. RESTRICTIVE MARKINGS	
2a. SECURITY CLASSIFICATION AUTHORITY		3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.	
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE			
4. PERFORMING ORGANIZATION REPORT NUMBER(S)		5. MONITORING ORGANIZATION REPORT NUMBER(S) AFVAL-TR-86-40006 Vol III, Part 7	
6a. NAME OF PERFORMING ORGANIZATION General Electric Company Production Resources Consulting	6b. OFFICE SYMBOL (If applicable) AFVAL/MLTC	7a. NAME OF MONITORING ORGANIZATION AFVAL/MLTC	
6c. ADDRESS (City, State and ZIP Code) 1 River Road Schenectady, NY 12345		7b. ADDRESS (City, State and ZIP Code) WPAFB, OH 45433-6533	
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Materials Laboratory Air Force Systems Command, USAF	8b. OFFICE SYMBOL (If applicable) AFVAL/MLTC	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER F33615-80-C-5155	
8c. ADDRESS (City, State and ZIP Code) Wright-Patterson AFB, Ohio 45433		10. SOURCE OF FUNDING NOS.	
		PROGRAM ELEMENT NO. 78011F	TASK NO. 62
		PROJECT NO. 7500	WORK UNIT NO. 01
11. TITLE (Include Security Classification) (See Reverse)			
12. PERSONAL AUTHOR(S) Marvin, Nancy			
13a. TYPE OF REPORT Final Technical Report	13b. TIME COVERED 22 Sept 1980 - 31 July 1985	14. DATE OF REPORT (Yr., Mo., Day) 1985 November	15. PAGE COUNT 29
16. SUPPLEMENTARY NOTATION ICAM Project Priority 6201		The computer software contained herein are theoretical and/or references that in no way reflect Air Force-owned or -developed computer software.	
17. COSAT CODES		18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)	
FIELD	GROUP	SUB GR.	
1305	0005		
19. ABSTRACT (Continue on reverse if necessary and identify by block number) <i>from Xerox</i> This administrative manual gives explicit instructions for creating VAX and IBM releases. General information for administering Software Configuration Management is also provided. ←			
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED <input checked="" type="checkbox"/> SAME AS RPT. <input type="checkbox"/> DTIC USERS <input type="checkbox"/>		21. ABSTRACT SECURITY CLASSIFICATION Unclassified	
22a. NAME OF RESPONSIBLE INDIVIDUAL David L. Judson		22b. TELEPHONE NUMBER (Include Area Code) 513-255-8876	22c. OFFICE SYMBOL AFVAL/MLTC

11. Title

Integrated Information Support System (IISS)
Vol III - IISS Configuration Management
Part 7 - SCM Administrator's Manual

Accession For	
NTIS GRA&I	<input checked="checked" type="checkbox"/>
DTIC TAB	<input checked="checked" type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	



→ The Integrated Information Support System is a test computing environment used to investigate and demonstrate and test the concepts of information management and information integration in the contexts of Aerospace Manufacturing. Specifically, IISS addresses the problems of integration of data resident on heterogeneous databases supported by heterogeneous computers, interconnected via a Local Area Network. A common Data Model is maintained and provides the mechanism required to integrate the data.

→ to 1473

PREFACE

This administrator's manual covers the work performed under Air Force Contract F33615-80-C-5155 (ICAM Project 6201). This contract is sponsored by the Materials Laboratory, Air Force Systems Command, Wright-Patterson Air Force Base, Ohio. It was administered under the technical direction of Mr. Gerald C. Shumaker, ICAM Program Manager, Manufacturing Technology Division, through Project Manager, Mr. David Judson. The Prime Contractor was Production Resources Consulting of the General Electric Company, Schenectady, New York, under the direction of Mr. Alan Rubenstein. The General Electric Project Manager was Mr. Myron Hurlbut of Industrial Automation Systems Department, Albany, New York.

Certain work aimed at improving Test Bed Technology has been performed by other contracts with Project 6201 performing integrating functions. This work consisted of enhancements to Test Bed software and establishment and operation of Test Bed hardware and communications for developers and other users. Documentation relating to the Test Bed from all of these contractors and projects have been integrated under Project 6201 for publication and treatment as an integrated set of documents. The particular contributors to each document are noted on the Report Documentation Page (DD1473). A listing and description of the entire project documentation system and how they are related is contained in document FTR620100001, Project Overview.

The subcontractors and their contributing activities were as follows:

TASK 4.2

<u>Subcontractors</u>	<u>Role</u>
Boeing Military Aircraft Company (EMAC)	Reviewer
D. Appleton Company (DACOM)	Responsible for IDEF support, state-of-the-art literature search
General Dynamics/ Ft. Worth	Responsible for factory view function and information models

CMA620124000
1 November 1985

Subcontractors

Role

**Illinois Institute of
Technology**

**Responsible for factory view
function research (IITRI)
and information models of
small and medium-size business**

North American Rockwell

Reviewer

Northrop Corporation

**Responsible for factory view
function and information
models**

Pritsker and Associates

Responsible for IDEF2 support

SofTech

Responsible for IDEFO support

TASKS 4.3 - 4.9 (TEST BED)

Subcontractors

Role

**Boeing Military Aircraft
Company (EMAC)**

**Responsible for consultation on
applications of the technology
and on IBM computer technology.**

**Computer Technology
Associates (CTA)**

**Assisted in the areas of
communications systems, system
design and integration
methodology, and design of the
Network Transaction Manager.**

**Control Data Corporation
(CDC)**

**Responsible for the Common Data
Model (CDM) implementation and
part of the CDM design (shared
with DACOM).**

**D. Appleton Company
(DACOM)**

**Responsible for the overall CDM
Subsystem design integration
and test plan, as well as part
of the design of the CDM
(shared with CDC). DACOM also
developed the Integration
Methodology and did the schema
mappings for the Application
Subsystems.**

CMA620124000
1 November 1985

Subcontractors

Role

Digital Equipment
Corporation (DEC)

Consulting and support of the
performance testing and on DEC
software and computer systems
operation.

McDonnell Douglas
Automation Company
(McAuto)

Responsible for the support and
enhancements to the Network
Transaction Manager Subsystem
during 1984/1985 period.

On-Line Software
International (OSI)

Responsible for programming the
Communications Subsystem on the
IBM and for consulting on the
IBM.

Rath and Strong Systems
Products (RSSP) (In 1985
became McCormack & Dodge)

Responsible for assistance in
the implementation and use of
the MRP II package (PIOS) that
they supplied.

SofTech, Inc.

Responsible for the design and
implementation of the Network
Transaction Manager (NTM) in
1981/1984 period.

Software Performance
Engineering (SPE)

Responsible for directing the
work on performance evaluation
and analysis.

Structural Dynamics
Research Corporation
(SDRC)

Responsible for the User
Interface and Virtual Terminal
Interface Subsystems.

Other prime contractors under other projects who have
contributed to Test Bed Technology, their contributing
activities and responsible projects are as follows:

Contractors

ICAM Project

Contributing Activities

Boeing Military
Aircraft Company
(BMAC)

1701, 2201,
2202

Enhancements for IBM
node use. Technology
Transfer to Integrated
Sheet Metal Center
(ISMC)

CMA620124000
1 November 1985

<u>Contractors</u>	<u>ICAM Project</u>	<u>Contributing Activities</u>
Control Data Corporation (CDC)	1502, 1701	IISS enhancements to Common Data Model Processor (CDMP)
D. Appleton Company (DACOM)	1502	IISS enhancements to Integration Methodology
General Electric	1502	Operation of the Test Bed and communications equipment.
Hughes Aircraft Company (HAC)	1701	Test Bed enhancements
Structural Dynamics Research Corporation (SDRC)	1502, 1701, 1703	IISS enhancements to User Interface/Virtual Terminal Interface (UI/VTI)
Systran	1502	Test Bed enhancements. Operation of Test Bed.

TABLE OF CONTENTS

		<u>Page</u>
SECTION 1.0	INTRODUCTION	1-1
SECTION 2.0	GENERAL DESCRIPTION OF SCM	2-1
2.1	Scope	2-1
2.2	Directory Structure	2-1
2.3	SCM Administrator Functions	2-3
2.4	Summary	2-3
SECTION 3.0	UPDATING THE SCM DATABASE	3-1
3.1	General Description	3-1
3.2	Steps to Use UPDCI	3-3
3.3	Steps to Use CVTNEW	3-4
SECTION 4.0	VAX RELEASE PROCEDURES	4-1
4.1	General Description	4-1
4.2	Steps For a VAX Release	4-3
SECTION 5.0	IBM RELEASE PROCEDURES	5-1
5.1	General Description	5-1
5.2	Steps for an IBM Release	5-2
SECTION 6.0	MANUAL SCM PROCEDURES	6-1

SECTION 1

INTRODUCTION

The SCM administrative functions are a set of automated procedures to be used when appropriate by the SCM administrator. The SCM administrator uses many VAX and IS/Workbench functions manually to accomplish various restructuring tasks such as moving files, creating new directories, deleting libraries, and so forth. The automated procedures are available to handle large tasks such as creating a release.

This document is divided into five more sections:

- Section 2 gives a general description of SCM, including specific information on all SCM directories.
- Section 3 provides information on updating the SCM database file, which contains records on all IISS files.
- Section 4 provides general information and specific directions for all stages of the creation of a VAX release, excluding the directions for testing the release.
- Section 5 provides the above information for an IBM release.
- Section 6 describes the most commonly used manual procedures.

SECTION 2

GENERAL DESCRIPTION OF SCM

2.1 Scope

IISS Software Configuration Management (SCM) has the following functions:

- Storing current source code while preserving the history of changes to it.
- Controlling changes to source code.
- Facilitating releases with automated functions.

The SCM system consists of Source Code Control System (SCCS), some DCL (Digital Command Language) code created by General Electric, and one C program used to interface between the DCL and SCCS. SCCS, a product of Interactive Systems, provides low-level functions for the storage and changing of source code. The SCCS functions are never seen by the general users of SCM; access to the functions is through the relatively user-friendly DCL functions.

2.2 Directory Structure

The directories used for SCM are [SIISS], [NIISS], [IISS], [IISSIBM], [CMDB], [IISSCM], and [TIISS]. The directory structure in all of these except [IISSCM] and [CMDB] is similar; under the top directory are the subsystems, and one level below that are the subdirectories. The following gives brief descriptions of the functions of these areas:

- [SIISS] is where almost all source code is stored. The storage of the source code is controlled by SCCS. Each source code file is given a prefix SX and is stored in a form that separately preserves all changes to the original file. Within an SCCS file, changes for a given release have a corresponding SCCS internal number (6 for Release 1.6, 10 for Release 2.0, 15 for Release 2.5, etc.). The files are created with the SCCS "admin" command and changed with the SCCS "delta" command, and any change level of the file can be printed into a file with the SCCS "get" command. Due

to the possible duplicate names of host-dependent code, there are separate directories (.VAX and .IBM) under all directories for storing the host-dependent code. These directories are only used in [SIISS]; during releases, the code for the proper host is pulled into the same directory as generic code.

- [WIISS] is where any source code is stored that cannot be put into SCCS. At one time it was used to store PLI object files, but is now used only for .FDL (Forms Definition Language) and .MSG (message) files.
- [IISS] is the VAX release directory. Prior to a release, most of its directories are cleared out so that the current source can be brought into it from [SIISS] for compilation and linking. [IISS.COM] is where the VAX release procedures are located. [IISS.BUILD] is the location of the .COM files that are used to build the release on a VAX.
- [IISSIBM] is the IBM release directory. It serves the same function as [IISS] except that it is for IBM releases. [IISSIBM.COM] is where the IBM release procedures are located. [IISSIBM.BUILD] is the location of the JCL files that are used to build the release on the IBM. Some of the JCL files are permanent, but the CL... and the LK... files must be created for each release to provide the JCL needed to compile and link on the IBM.
- [CMDB] is the SCM database directory. It contains CI.DAT, the indexed sequential file containing information on all source code. The accuracy of CI.DAT is very important since it controls which code goes where into releases. [CMDB] contains NEWITEM.DAT, listing all new files that need to go into CI.DAT, and RETURN.DAT, listing all files that were changed since the previous release. It also contains various cross-reference and data files used for the SCM user functions. Directories under CMDB are used to store information used for particular releases. For example, [CMDB.RELO18] contains the command files created for release 1.8, and [CMDB.SXRELO18] contains SCCS source files that were deleted from SCM around the time of release 1.8. These release-specific directories are stored on disk for a couple of releases and then are moved to tape

and archived.

- [IISSCM] is the production directory for the CM user functions, such as NEWITEM and CHECKOUT.
- [TISS] is the test directory for releases. All executables, command files, data files, message files, and forms files are moved to this directory (mostly to [TISS.TESTXX]) for integration testing and debugging of a release. Upon completion of the testing, appropriate code is moved to the production directory [PISS] and the development directory [DISS].

2.3 SCM Administrator Functions

The SCM Administrator is responsible for maintaining all SCM code, including both user functions and administrative functions. The administrator is also responsible for running administrative functions, both for updating the SCM database and creating releases. The database is always updated at the beginning of each release with all newitems, but throughout a release new files may be added. At any time file attributes (e.g. host, subdirectory, or link parameters) may be changed, or obsolete files may need to be deleted. When files are deleted or when certain file attributes are changed, the source file under [SISS] must be moved. This moving is done automatically by the CI.DAT update program (UPDCI.COM), which must be executed while logged onto [SISS], due to the protections under [SISS].

2.4 Summary

The SCM functions provide a relatively user-friendly interface to SCCS code, provide the desired protections and functionalities, and provide much automation of release procedures. Functionalities provided include the ability to use SCM across disks, across different UIC groups, and from directories lacking world access. In order to provide the latter two functions, it was necessary to use the VAX Install utility to permanently install the SCCS interface executable (INTER.EXE) with DETACH and SYSNAM privileges and to permanently install the SCCS executables DELTA.EXE and ADMIN.EXE with SYSPRV privilege.

The SCM user functions are described separately in the SCM User's Manual, CMA620124001. The administrative functions are described in the following sections.

SECTION 3

UPDATING THE SCM DATABASE

3.1 General Description

The most important file in SCM is one that keeps track of all current files that are part of IISS. This database file must be updated when any file is added to the system or deleted from the system or when any file has various information for it changed. The database file is CI.DAT, an indexed sequential file with each record providing attribute fields for a given file. These attribute fields provide the information needed during a release so that the file will be moved to the proper directory, so that the file will be compiled and linked as needed, and so that the object file will be put into the proper library. The CI.DAT file is copied at the end of each release and is provided to all developers as a reference. The alphabetical and sorted versions are Supplement 1 and Supplement 2 to the SCM User's Manual, CMU620124001. The fields in the file are the following:

- file (8 characters, although 7 characters is the maximum allowable filename length)
- extension (3 characters)
- host (1 character)
- subsystem (5 characters)
- subdirectory (10 characters)
- documentation group (10 characters)
- common object library (7 characters)
- link command file (12 characters)
- linking parameters (40 characters)

The primary key for the CI.DAT file is the file and extension. Therefore these two fields cannot be altered in a given record. If either file or extension need to be changed for a given file, the file must be deleted from SCM and then newitened with the new name. The alternate key is file,

extension, and host. This alternate key is used for accessing a unique record, and duplicate entries are not allowed.

The two functions that are used to update CI.DAT are UPDCI.COM and CVTNEW.COM. UPDCI is used to alter a given file record or to delete it. CVTNEW is used to add new files that have been newitemed to the database.

UPDCI prompts repeatedly for a filename (FILE.EXT) and allows for either updating the file or deleting it. Pressing <RETURN> at the filename prompt exits the program. If the administrator is updating a record, he is prompted for the changes to each field. Pressing <RETURN> at any prompt keeps that field unchanged. If any fields are changed so that the [SIISS] file must be moved (such as subdirectory or host), that move takes place automatically. If any records are deleted, the [SIISS] file is moved to an obsolete file location under [CMDB], so that the name can be reused if desired. Because of the [SIISS] file moves, it is necessary to be logged onto [SIISS] to run the UPDCI command file.

CVTNEW is called automatically by VSTART.COM at the beginning of a release. It converts the newitems in NEWITEM.DAT that are not designated for future releases into the correct format for being added to CI.DAT and then adds them. The files in NEWITEM.DAT that are designated for future releases remain in the NEWITEM.DAT file. CVTNEW can be invoked separately for adding any additional newitems that are added during a release. It is called with one parameter designating the current release (6 for Release 1.6, 10 for Release 2.0, 15 for Release 2.5, etc.).

3.2 Steps to Use UPDCI

1. Log onto [SIISS]
2. Set your default to [IISS.COM]
3. \$ @UPDCI
4. Enter FILE.EXTENSION and HOST at the prompts
5. View the file information
6. Answer whether you want to update the file information (default is NO)

7. If (6) is NO, answer whether you want to delete the file (default is NO)
8. If (6) is YES, type in answers to the fields you want changed. Pressing 'RETURN' at any prompt keeps that field unchanged. If you want the field changed to a blank field, type in BLANK.
9. The file will then be updated and you will be prompted for another file and extension. Press 'RETURN' when you have finished updating files and wish to exit the program.

3.3 Steps to Use CVTNEW

1. Log onto [IISS]
2. Since CVTNEW will delete the most recent version of NEWITEM.DAT, you should save it in the release directory by:

\$ copy [CMDB]file.extension [CMDB.RELOxx]*

where xx is the two digits in the release number (18 for Release 1.8).
3. Set your default to [IISS.COM]
4. \$ CVTNEW y

where y is the SCCS internal number that corresponds with the release number (6 for Release 1.6, 10 for Release 2.0, 15 for Release 2.5, etc.).
5. Check the file which will automatically be sent to the printer. This file lists any exceptions that could not be added to CI.DAT.

SECTION 4

VAX RELEASE PROCEDURES

4.1 General Description

A release is a step-by-step sequential procedure that must be checked at each stage to assure that there are no problems. The release account, where the VAX release is done, is IISS. The order that subsystems are released is: (1)IPC, (2)ERRLOG, (3)NTM, (4)COMM, (5)UI, and (6)CDM. A VAX release includes the following:

- The CI.DAT file is updated with the addition of all new files (newitems), the deletion of all obsolete files, and the changing of any CI.DAT information as needed.
- The CI.DAT file is sorted by subsystem and subdirectory in order to build command files for the release.
- Command files for the release are built, one subsystem at a time.
- IISS directories are emptied and empty object libraries are created.
- Source code is moved into IISS.
- All compiles, library replaces, and links are done, one subsystem at a time.
- Forms are created.
- The files needed for running IISS are moved to [TIISS], the test directory.
- The testing procedure includes doing precompiles and moving the resultant COBOL files to IISS. These files and the command files created to compile and link them become part of the release.
- When testing is complete, the command files needed to build a release are moved to the [IISS.BUILD] directory and release tapes are made.

These general steps are normally repeated many times, at least partially, for a release. When a problem is encountered, the steps must either be repeated or the problem area must be manually handled to solve the problem. For instance, if a batch job to compile the User Interface is run and three of the files do not compile correctly, the problem files could be fixed and manually compiled. On the other hand if an include file needed changing and if it were included in twenty or thirty files, it would probably be easiest to fix the include file and then redo the entire compile batch job. Backtracking is often required during the release. For instance, if a file in the NTM services needs to be modified to solve problems encountered in NTM testing, all links that use the services must be redone. When source code is changed to solve a problem encountered during testing, backtracking must be done as far back as necessary to fix anything affected by the change.

4.2 Steps for a VAX Release

1. Log onto IISS and set your default to [IISS.COM]. This is where the administrative release procedures are located.

2. **\$ @VSTART**

This procedure can be run only at the beginning of each release. It uses the current release number from [CMDB]RELNUM.DAT to create a release specific directory under [CMDB] for the release, e.g. [CMDB.REL014] for release 1.4. The current NEWITEM.DAT and RETURN.DAT are copied into this directory. CVTNEW.COM is called to update CI.DAT with all newitems for the releases. Other newitems are retained for future releases. When newitems or returns are added in the middle of the release, parts of this procedure must be done manually. That is, NEWITEM.DAT and RETURN.DAT are copied into the release directory, CVTNEW.COM is called, and RETURN.DAT is deleted from [CMDB].

3. If any items in CI.DAT need to be deleted or modified, run UPDCI.COM as described in Chapter 3.
4. **\$ @VSUBSYS**

This procedure creates separate files in the format of CI.DAT for the subsystems. These files are .TMP files and are put in the release specific directory, [CMDB.RELOxx]. For example, IPC.TMP is a sorted list of the IPC subsystem.

5. Build the command files for all the subsystems:

```
$ @VELDCOM IPC
$ @VELDCOM ERR
$ @VELDCOM NTH
$ @VELDCOM COMM
$ @VELDCOM UI
$ @VELDCOM CDM
```

All command files to release the subsystems are created. The files created are put into the [CMDB.RELOxx] directory and are keyed to the subsystem specified. For example, for the IPC subsystem, the files created are the following:

- GTIPC.COM will get (retrieve) IPC source code from SISS and NISS, putting it in IISS. Include files are copied to the appropriate IISS directories (.SOURCELIB for .INF and .INC files, and .HLIB for .H files).
- ILIPC.COM will do library replaces of .INC files to the include text library, [IISS.SOURCELIB]IISSCLIB.TLB.
- CLIPC.COM will do the compiles.
- OLIPC.COM will do object library replaces and then delete the .OBJ files.
- LKIPC.COM will do the links.

6. \$ @VCREGET

This creates a file, DOGET.COM, that will retrieve the source code for all subsystems.

7. Now, create the files that will do compiles, library replaces, and links for the subsystems:

```
$ @VCREDO IPC
$ @VCREDO ERR
$ @VCREDO NTH
$ @VCREDO COMM
$ @VCREDO UI
$ @VCREDO GDM
```

One file is created for each subsystem and is put into the [CMDB.RELOxx] directory. The file name is keyed to the subsystem specified. For example, DOIPC.COM is created for the IPC subsystem.

8. \$ @VDELALL

This procedure calls VDELETE for all subsystems to clean out all files from the IISS directories. It also deletes the forms library, include libraries, and the common object library, IPCLIB. All libraries are recreated empty.

9. Set your default to [CMDB.RELOxx] and start the batch job:

```
$ submit/notify DOGET.COM
```

This batch job will pull all source code for all subsystems into IISS and copy include files to the include libraries. This will take hours to complete. When it is finished a batch output log will be printed. Check the log for any errors, and correct them before proceeding.

10. Set your default back to [IISS.COM] to do the following:

```
$ @VINIT
```

This procedure does some of the ERRLOG compiles and library replaces. These must be done prior to building the IPC subsystem.

11. Now the various subsystems can be built. Running VDORUN creates a batch job that does compiles, library replaces, and links for a given subsystem. The batch jobs will take various lengths of time depending upon the size of the subsystem. These runs must be done in the following order because of links that depend on

compilations of other subsystems:

```
$ SET DEF [IISS.COM]
$ @VDORUN IPC
$ @VDORUN ERR
$ @VDORUN NTM
$ @VDORUN COMM
$ @VDORUN UI
$ @VDORUN CDM
```

A batch output log will be printed at the end of each batch job. Check it for errors. Some links in the UI and the CDM will not work at this time because they depend upon precompiled modules. These will be done manually at a later time.

12. The forms for the User Interface must now be created by running the stand-alone version of FLAN to create all the .FD files from the .FDL files. The .FDL files are located in [IISS.FD], and that is where the forms will be created. Set your default to [IISS.FD] and copy the flan executable to there:

```
$ copy [IISS.UI.FLAN]FLANSA.EXE *
```

You need to reassign the IISSULIB logical:

```
$ assign [IISS.FD] IISSULIB
```

Make a list of all .FDL files in the directory. Run FLANSA once for each .FDL file. When you run FLANSA, at the args prompt enter the name of the .FDL file, and all .FD files defined in that file will be created.

13. Now the release is basically built, and testing and precompiling need to be done. (Testing can actually be started after the NTM has been built, and other subsystems can be tested as they are built.) The testing is done in the TIISS account, under [TIISS.TESTXX]. After the directories in TIISS have been cleaned out, log onto IISS, set your default to [IISS.COM], and move the necessary files to TIISS:

```
$ @MOVEALL
```

14. Directions for integration testing are found in other

manuals. (See IISS Documentation Description, CMB620100000, for a list of the documents.) Normally the IPCs, ERRLOG, and COMM are tested first. Then the NTM tests are run. Then the UI tests that are not dependent upon a precompiled module are run. Then, at the beginning of CDM testing, the precompiles needed for the rest of the release build are done. Documentation for this is provided for each release by subcontractors responsible for the subsystems requiring precompilation. Most information on this is contained in an attachment to the CDMP Test Case Report Documentation, called Define/Construct NDDL.

15. When all precompilations needed to finish building IISS have been completed, the .TMP files created during precompilation must be moved back to IISS for the build. Move these files to [IISS.CDM.NDDL].
16. Log back onto IISS and set your default to [.CDM]. Now execute the procedures BLDNDDL.COM and GRELLST.COM as described in the manual, Define/Construct NDDL, to create the procedures to do the needed compiles and links. After you have run the compile and link procedures, you can do the links that failed earlier due to references to precompiled modules. Now IISS is entirely built, and testing of the UI and the CDM can be completed.
17. When testing is deemed complete and all required changes have been made to SCM, IISS, and [CMDB.RELOxx], IISS can be prepared for making the release tape. Directories, such as [IISS.NTM...] and [IISS.UI...] should be purged, and .LIS files and .LOG files should be deleted.
18. The .BUILD directory must be made current. The following procedures are done manually:
 - Copy the CL..., OL..., and LK... files over into [IISS.BUILD] from [CMDB.RELOxx].
 - Copy the compile and library replace command files that were created in [IISS.CDM] for the precompiled modules together into [IISS.BUILD]CMPNDDL.COM.
 - Copy the link file that was created in [IISS.CDM]

for the precompiled modules into
[IISS.BUILD]LNKLST.COM.

- Check all command files in [IISS.BUILD], such as MOVERUN.COM, to see if they are current and correct.
- Now purge the files in [IISS.BUILD].

19. The [IISS.ORACLE] directory needs to be updated for the new release. Your ORACLE database administrator should provide you with current .BCK and .VEW files from an export of the database, and he should verify the current correctness of ORAUSER.COM and ORAUSERL.COM.

20. The [IISS.RUNAREA] directory should contain only two files, SYSGEN.DAT and the .; file. The .; file should contain:

args:
args:

21. The release tape can now be made. Log onto IISS on the hard copy terminal and mount the tape. Then initialize the tape:

```
$ INIT/DENS=1600 MT: IISS  
$ MOUNT/FOR MT:
```

Now create the release tape:

```
BACKUP/VER/LIST-TP.LIS IISS_CM:[IISS...]  
/EXCLUDE=(*.OBJ,*.OLB,*.EXE,*.MAI,[IISS.COM]*.*,  
[IISS.UTILITY]*.*,[IISS.MCMM]*.*) MT:IISS.BCK
```

When the tape is finished, print the listing:

```
$ PRINT TP.LIS
```

22. The Installation Guide (OM 620124001) should be updated to be provided with the release tape.

23. It is recommended that a trial build be done with the tape on a VAX with the appropriate hardware and software. If any problems are encountered in

CMA620124000
1 November 1985

the build, the appropriate corrections may be made in [IISS] prior to the making of another release tape. Ideally this procedure is repeated as necessary until a tested final version of the tape encounters no problems.

24. Since the release is complete, the release number should be updated in RELNUM.DAT by entering the following (with your default set to [IISS.COM]):

\$ EVEND reln

where reln is the next release number (e.g. 2.1)

SECTION 5

IBM RELEASE PROCEDURES

5.1 General Description

IBM release tapes are created from an IBM release account on the VAX, IISSIBM. The directory structure of IISSIBM is similar to that of IISS, the VAX release directory. Code that is used for building the release on an IBM is located in [IISSIBM.BUILD], and release procedures that are used for creating the release tape from the VAX are located in [IISSIBM.COM].

The IBM release includes the following:

- VAX release procedures are used as needed to update CI.DAT and create the .TMP files that are the sorted lists for each subsystem.
- The IBM release directory is cleaned up in preparation for the new release.
- Command files for the release are built, one subsystem at a time.
- Source code is moved into IISSIBM.
- JCL files to do compiles and links on the IBM are moved to the [.BUILD] directory.
- A directory listing of all files to go on the tape is created.
- The release tape is created.
- The contents of the tape must be loaded onto an IBM to test the build procedure and to test the IBM version of IISS.

5.2 Steps for an IBM Release

1. Do steps 1,2,3, and 4 of the VAX release procedures (Section 4.2) as needed, unless they have already been done for the VAX release.

2. Log onto IISSIBM and set your default to [IISSIBM.COM]. To do the necessary prerelease cleanup, run the following:

```
$ @IDELALL
```

3. Build the command files to release the subsystems:

```
$ @IBLDCOM IPC  
$ @IBLDCOM NTM  
$ @IBLDCOM COMM  
$ @IBLDCOM UI  
$ @IBLDCOM CDM  
$ @IBLDCOM CICS
```

4. Pull the source code into IISSIBM:

```
$ SUBMIT/NOTIFY GTIPCxx  
$ SUBMIT/NOTIFY GTNTMxx  
$ SUBMIT/NOTIFY GTCOMMxx  
$ SUBMIT/NOTIFY GTUIxx  
$ SUBMIT/NOTIFY GTCDMxx  
$ SUBMIT/NOTIFY GTCICSxx
```

where xx is the release number (23 for Release 2.3). Check the logs that are automatically printed at the end of the batch jobs. These batch jobs may be run concurrently.

5. Move all the compile and link files to the [.BUILD] directory:

```
$ copy CL*.JCL [IISSIBM.BUILD]*  
$ copy LK*.JCL [IISSIBM.BUILD]*
```

6. While your default is still set to [IISSIBM.COM], it is necessary to create the log datasets:

```
$ @IBLDLOG
```

7. The tape can now be created. Log onto IISSIBM on the hard copy terminal and set your default to [IISSIBM.COM]. Mount a tape and then start creating it:

```
$ @CRTTAPE
```

CNA620124000
1 November 1985

When the tape is finished, print the listing:

\$ PRINT TAPE.MST

8. The Installation Guide (ON 620124002) should be updated to be provided with the release tape. Building and testing must be done on the IBM, and any needed corrections must be made on the VAX prior to making a new tape.

SECTION 6

MANUAL SCM PROCEDURES

If everything in a release were to work perfectly the first time, it would only be necessary to use the automated procedures. However this is rarely the case, so that knowing how to do parts of the procedures manually is a necessity.

When a file needs to be changed for any reason after the release is started, the new version can be pulled into IISS. The file must be changed in Configuration Management by using the user functions CHECKOUT and RETURN. This is normally the responsibility of the developer in charge of the particular subsystem. Then you must log onto IISS and set your default to the directory where the changed file resides. Delete the old file. Then bring in the new version with the following SCCS command:

```
% GET -Rn IISS_CM-/SISS/subsys/subdir/SXfilename.extension
```

Fill in the appropriate values for subsystem, subdirectory, filename, extension, and n (the SCCS internal number, e.g. 10 for Release 2.0, 15 for Release 2.5, or 29 for Release 3.9).

When a new file has been pulled in, all needed compiles, library replaces, and links must be performed to accommodate the change into the release. These are accomplished with standard VMS procedures. Refer to the appropriate Product Specification(s) for information on what will be affected by the changed module or include file. The format of some commonly used procedures is as follows:

```
% COBOL/ANSI/NOLIST filename
```

```
% FORTRAN/NOLIST filename
```

```
% CCX filename
```

```
% LIBRARY/REPLACE xOLB.OLB filename.OBJ
```

where x is the first 4 characters of the directory.

```
% COPY filename.H [IISS.HLIB]*
```

```
% COPY filename.INC [IISS.SOURCELIB]*
```

CMA620124000
1 November 1985

\$ LIBRARY/REPLACE/TEXT IISSCLIB filename.INC

\$ COPY filename.INF [IISS.SOURCELIB]

Deletes and purges should be done throughout as needed to keep IISS current and clean.

END

7-87

DTIC